CLAIMS

1. A method for transferring objects between a first (20, resp. 30) and a second (30, resp. 20) platform, said platforms (20, 30) providing a reflection framework adapted to allow introspection of the meta-information that describes the structure of said objects, characterized in that it includes the step of translating said objects to be transferred into XML payloads by reading (102) the characteristics of said objects through said reflection framework, thereby carrying out introspection of the meta-information that describes the structure of the classes of said objects.

2. The method of claim 1, characterized in that it includes the step of transferring said objects translated into XML payloads from said first platform (20, resp. 30) to said second (30, resp. 20) platform by means of an asynchronous delivery facility.

3. The method of claim 2, characterized in that it includes the step of re-translating said objects transferred as XML payloads to said second platform (30, resp. 20) into objects of said second platform (30, resp. 20).

4. The method of claim 1, characterized in that said step of translating includes the operations of:

- writing a start document (102),
- reading the respective objects properties (102) through said reflection framework, and
- writing the property name and value (102) if said property is basic type,

- writing the property name and calling recursively (104) if said property is object type, and

- writing the property name and value of all components (105) if said property is array type.

5. The method of claim 3, characterized in that said step of re-translating includes the operations of:

- resolving a class name from a document name (201)

- creating an object of said second platform and pushing it on a stack (202), and

- either pushing the XML content on stack (204) or pushing a new object on stack (205) after getting the property type of object on stack (203) or

- popping a value and assigning it to a top object (206)

6. The method of claim 1, characterized in that it includes the step of selecting at least one of said first and second platforms (20, 30) as a Java platform.

7. The method of claim 1, characterized in that it includes the step of selecting at least one of said first and second platforms (20, 30) as a .Net platform.

8. The method of claim 2, characterized in that it includes the step of selecting said asynchronous delivery facility as a peer-to-peer delivery facility.

9. The method of claim 2 or 8, characterized in that it includes the step of selecting TCP as said asynchronous delivery facility.

10. The method of claim 2 or 8, characterized in that it includes the step of selecting UDP as said asynchronous delivery facility.

11. The method of claim 2, characterized in that it includes the steps of providing in said second platform (30, resp. 20):

- at least one connector of said delivery facility as well as a set of worker threads (WT) for carrying out the requested operations, and
- a reactor function (35) for decoupling said at least one connector (36) from said set of worker threads (WT).

12. A system for transferring objects between a first (20, resp. 30) and a second (30, resp. 20) platform, said platforms (20, 30) providing a reflection framework adapted to allow introspection of the meta-information that describes the structure of said objects, characterized in that it includes a translator module (XmlObjectWriter) configured for translating said objects to be transferred into XML payloads by reading (102) the characteristics of said objects through said reflection framework, thereby carrying out introspection of the meta-information that describes the structure of the classes of said objects.

13. The system of claim 12, characterized in that it includes an asynchronous delivery facility for transferring said objects translated into XML payloads from said first platform (20, resp. 30) to said second (30, resp. 20).

14. The system of claim 13, characterized in that it includes a re-translator module (XmlObjectWriter) configured for re-translating said objects transferred as XML payloads to said second platform (30, resp. 20) into objects of said second platform (30, resp. 20).

15. The system of claim 12, characterized in that said translator module (XmlObjectWriter) is configured for performing the operations of:

- writing a start document (102),
- reading the respective objects properties (102) through said reflection framework, and
- writing the property name and value (102) if said property is basic type,
- writing the property name and calling recursively (104) if said property is object type, and
- writing the property name and value of all components (105) if said property is array type.

16. The system of claim 14, characterized in that said re-translator module (XmlObjectWriter) is configured for performing the operations of:

- resolving a class name from a document name (201)
- creating an object of said second platform and pushing it on a stack (202), and
- either pushing the XML content on stack (204) or pushing a new object on stack (205) after getting the property type of object on stack (203) or
- popping a value and assigning it to a top object (206)

17. The system of claim 12, characterized in that at least one of said first and second platforms (20, 30) is a Java platform.

18. The system of claim 12, characterized in that at least one of said first and second platforms (20, 30) is a .Net platform.

19. The system of claim 13, characterized in that said asynchronous delivery facility is a peer-to-peer delivery facility.

20. The system of claim 13 or 19, characterized in that said asynchronous delivery facility is TCP.

21. The system of claim 13 or 19, characterized in that said asynchronous delivery facility is UDP.

22. The system of claim 13, characterized in that said second platform (30, resp. 20) includes:

- at least one connector of said delivery facility as well as a set of worker threads (WT) for carrying out the requested operations, and
- a reactor function (35) for decoupling said at least one connector (36) from said set of worker threads (WT).

23. A computer program product loadable in the memory of at least one computer and including software code portions for performing the steps of the method of any of claims 1 to 11.